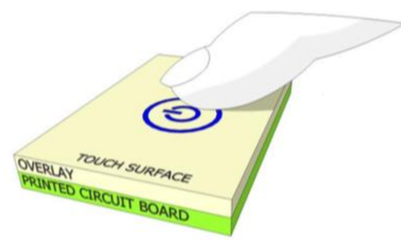
Exhibit 5

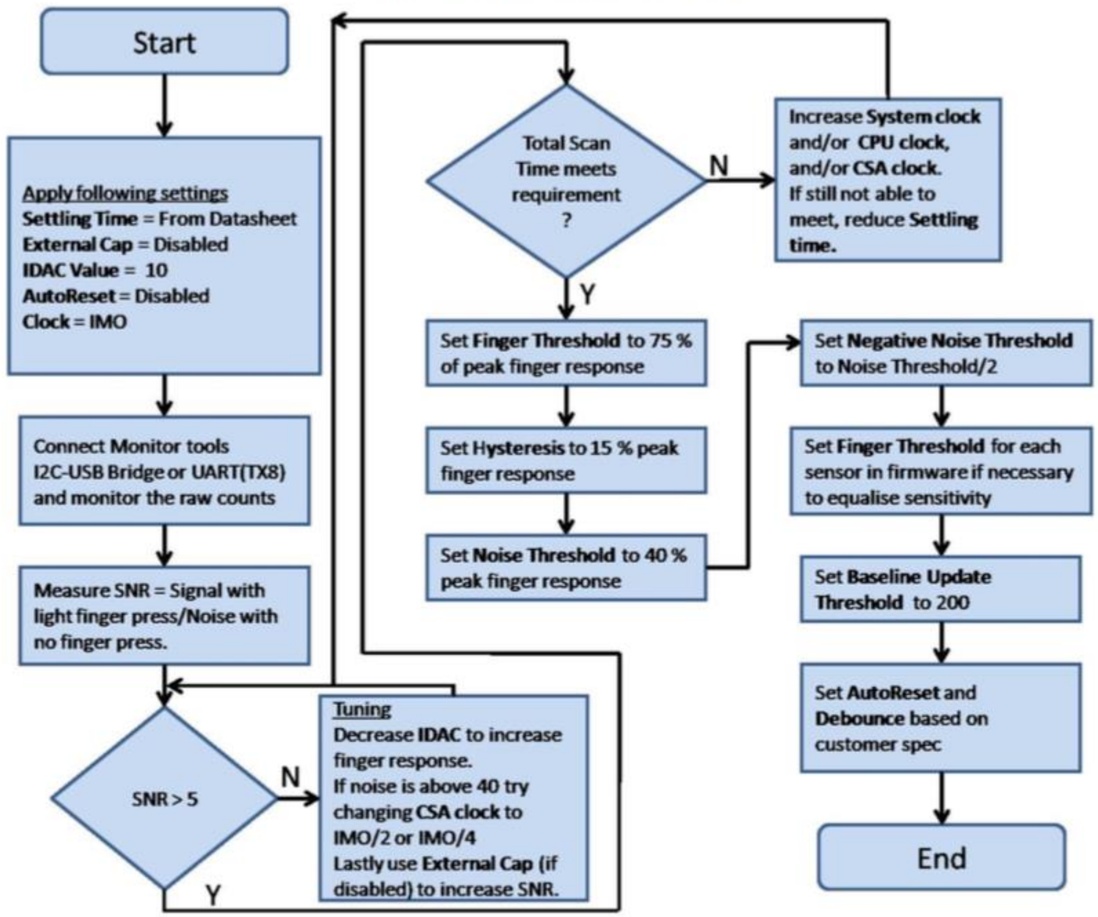
U.S. Patent No. 8,253,706 ("'706 Patent")

Exemplary Accused Product

Cypress products, including at least each of the following products (and their variations) infringe at least Claim 1 of the '706 Patent: Capsense enabled Cypress products, including MBR3, CY8CMBR2110, CY8CMBR2044, CY8CMBR2016, CY8CMBR2010, CY8CMBR3XXX, and Capsense-enabled PSoC. The infringement chart below is based on the Cypress PSoC with CapSense ("CapSense"), which is exemplary of the infringement of the '706 Patent.

Claim	CapSense
[1pre] A method comprising:	The CapSense touchcontroller provides capacitive touch sensing functionality, including in noisy and moist environments. For example, a CapSense-enabled touch sensor may exhibit noise sources impacting signal to noise (SNR) parameters. CapSense utilizes capacitor raw capacitor charge counts to distinguish touch from noise. Cypress' CapSense controllers use changes in capacitance to detect the presence of a finger on or near a touch surface, as shown in Figure 2-1. This touch button example illustrates a capacitive sensor replacing a mechanical button. The sensing function is achieved using a combination of hardware and firmware. See the Glossary for the definitions of CapSense terms. Figure 2-1. Illustration of a Capacitive Sensor Application  *See* Getting Started with CapSense, at p. 10, https://www.cypress.com/file/41076/download

	Firmware is a vital component of the CapSense system. It processes the raw count data and makes logical decisions. The amount of firmware development required for your application depends on which CapSense controller family you select. *See* Getting Started with CapSense, at p. 12, https://www.cypress.com/file/41076/download Optimal CapSense system performance depends on the board layout, button dimensions, overlay material, and application requirements. In addition to these factors, switching frequency and threshold levels must be carefully selected for robust and reliable performance. Tuning is the process of determining the optimum values for these parameters. Tuning is required to maintain high sensitivity to touch and to compensate for process variations in the sensor board, overlay material, and environmental conditions. Many of the CapSense devices support SmartSense, Cypress' Auto-tuning algorithm, which automatically sets parameters for optimal performance and continuously compensates for system, manufacturing and environmental changes. See SmartSense Auto-Tuning for more information. *See* Getting Started with CapSense, at p. 18, https://www.cypress.com/file/41076/download PSoC uses Cypress patented capacitive touch sensing methods known as CapSense Sigma Delta (CSD) for self-capacitance sensing and CapSense Crosspoint (CSX) for mutual-capacitance scanning. The CSD and CSX touch sensing methods provide the industry's best-in-class Signal-to-Noise Ratio. These sensing methods are a combination of hardware and firmware techniques. *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 15, https://www.cypress.com/file/46081/download Tuning the touch sensing user interface is a critical step in ensuring proper system operation and a pleasant user experience. The typical design flow involves tuning the sensor interface in the initial design phase, during system integration, and finally production fine-tuning before the production ramp. Because tuning is an iterative process, it can be time-consuming. SmartSense Auto-Tuning helps to simplify the user interface development cycle. In addition, the method is easy to use and reduces the design cycle time by eliminating the tuning process throughout the product development cycle, from prototype to mass production. *See* Getting Started with CapSense, at p. 20, https://www.cypress.com/file/41076/download

	Figure 8. CSA Calibration Flowchart  *See* CSA Software Filters with Ezl2Cs Slave on CY8C20xx6, at p. 10, https://www.cypress.com/file/137541/download

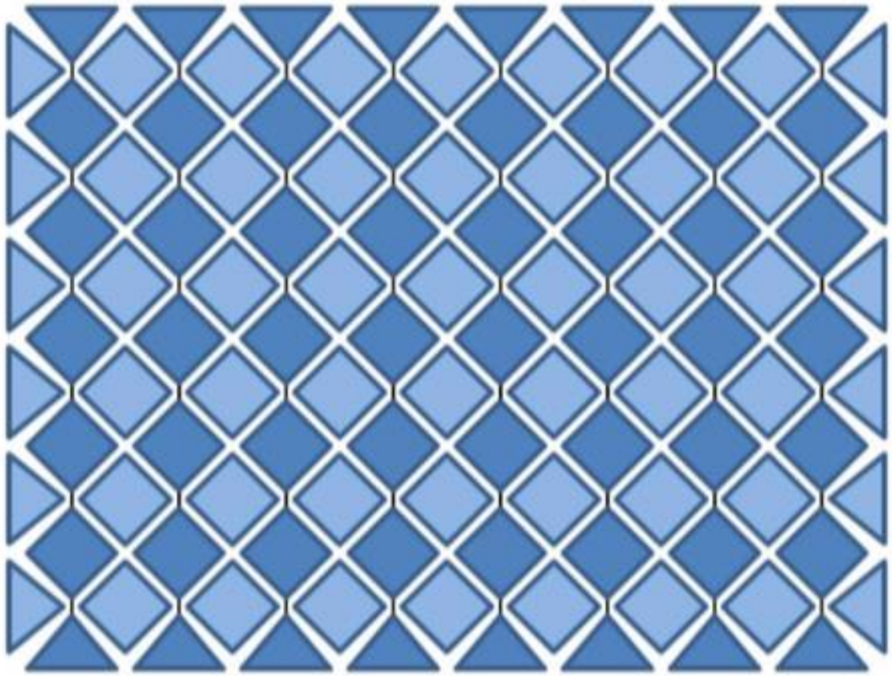
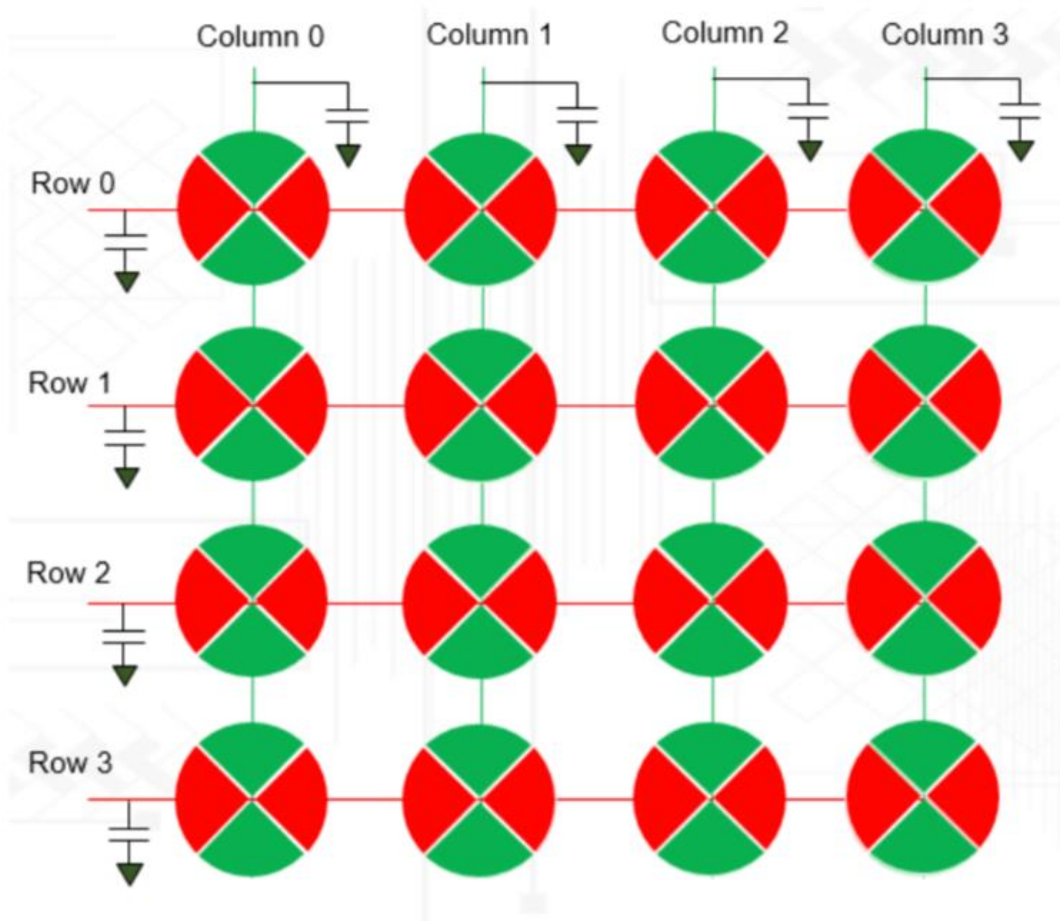
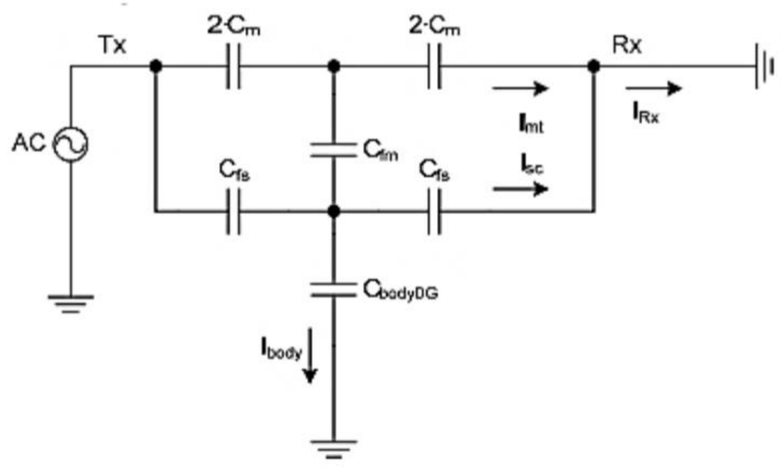
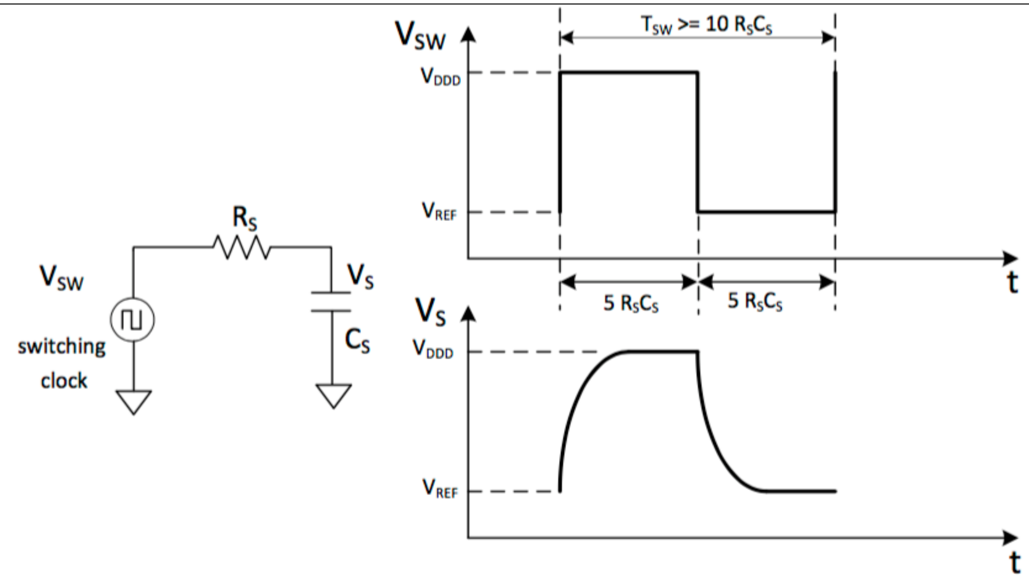
[1a] acquiring a first response from an output line of a matrix touch screen, the first response being a capacitively induced signal derived from a rising edge of a pulse applied to an input line of the matrix touch screen;	The CapSense touchcontroller acquires a first response from an output line of a matrix touch screen, the first response being a capacitively induced signal derived from a rising edge of a pulse applied to an input line of the matrix touch screen. For example, CapSense-enabled touch sensor acquires a first signal from the matrix touch sensor during the rising edge of the switching clock. Figure 2-27. Trackpad Sensor Arrangement  *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 27, https://www.cypress.com/file/41076/download

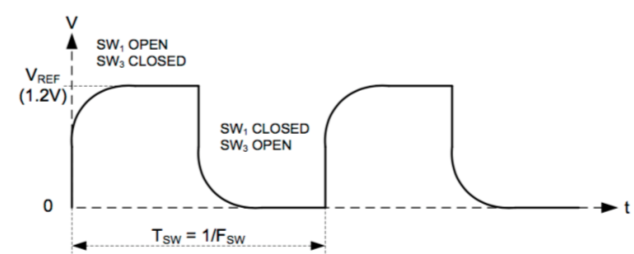
	Figure 2-14. Matrix Buttons Based on CSD  *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 25, https://www.cypress.com/file/41076/download

	In a mutual-capacitance measurement system, a digital voltage (signal switching between V_{DDD} and GND) is applied to the TX pin and the amount of charge received on the RX pin is measured. The amount of charge received on the RX electrode is directly proportional to the mutual capacitance (C_M) between the two electrodes. *See* Getting Started with CapSense, at p. 14, https://www.cypress.com/file/41076/download Figure 7-34. Equivalent Circuit of the CSX Sensor when Finger Is Placed on the Button  *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 135, https://www.cypress.com/file/46081/download



See PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 19, https://www.cypress.com/file/41076/download

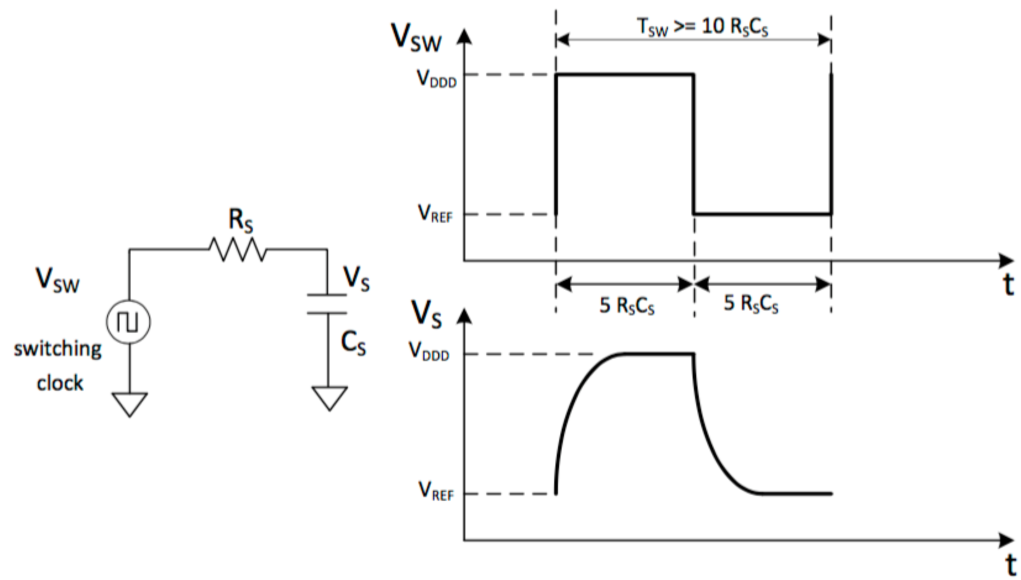
Figure 3-5. Voltage across Sensor Capacitance



Equation 3-2 gives the value of average current taken from AMUXBUS A.

Equation 3-2. Average Current Sinked from AMUXBUS A to GPIO through CapSense Sensor (I_{CS})

$$I_{CS} = C_S F_{SW} V_{REF}$$

	See PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 32, https://www.cypress.com/file/46081/download
[1b] acquiring a second response from the output line of the matrix touch screen, the second response being a capacitively induced signal derived from a falling edge of the pulse applied to the input line of the matrix touch screen; and	The CapSense touchcontroller acquires a second response from the output line of the matrix touch screen, the second response being a capacitively induced signal derived from a falling edge of the pulse applied to the input line of the matrix touch screen. For example, CapSense senses the sourced current I_{CS} sourced from the capacitor the sensor capacitor.  *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 45, https://www.cypress.com/file/41076/download

	The sigma-delta converter can operate in either IDAC sourcing mode or IDAC sinking mode. ■ IDAC Sourcing Mode: In this mode, the GPIO Cell Capacitance to Current Converter sinks current from CMOD through AMUXBUS A, and the IDACs then source current to AMUXBUS A to balance its voltage. ■ IDAC Sinking Mode : In this mode, the GPIO Cell Capacitance to Current Converter sources current from CMOD to AMUXBUS A and the IDACs sink current through AMUXBUS A to balance its voltage. In both the above-mentioned modes, the sigma delta converter can operate in either single IDAC mode or dual IDAC mode: ■ In the single IDAC mode, the modulation IDAC is controlled by the sigma-delta converter; the compensation IDAC is always OFF. ■ In the dual IDAC mode, the modulation IDAC is controlled by the sigma-delta converter; the compensation IDAC is always ON. *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 34, https://www.cypress.com/file/46081/download In the dual IDAC mode, the compensation IDAC is always ON. If I_{COMP} is the compensation IDAC current, the equation for the raw count in the IDAC Sourcing mode is: Equation 3-8. Dual IDAC Sourcing Raw Count $$\text{raw count} = (2^N - 1)\frac{V_{REF}\,F_{SW}}{I_{MOD}}C_S - (2^N - 1)\frac{I_{COMP}}{I_{MOD}}$$ Raw count in the IDAC Sinking mode is given by Equation 3-9. Equation 3-9. Dual IDAC Sinking Raw Count $$\text{raw count} = (2^N - 1)\frac{(V_{DD} - V_{REF})\,F_{SW}}{I_{MOD}}C_S - (2^N - 1)\frac{I_{COMP}}{I_{MOD}}$$ *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 34, https://www.cypress.com/file/46081/download

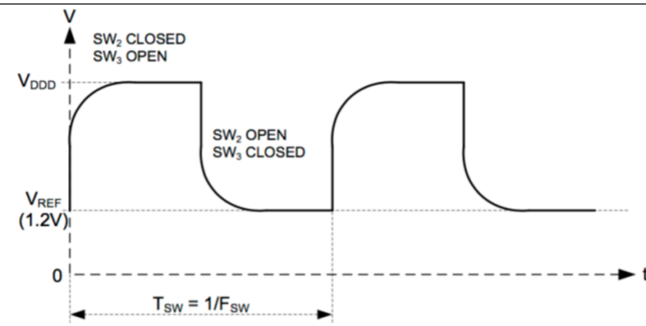
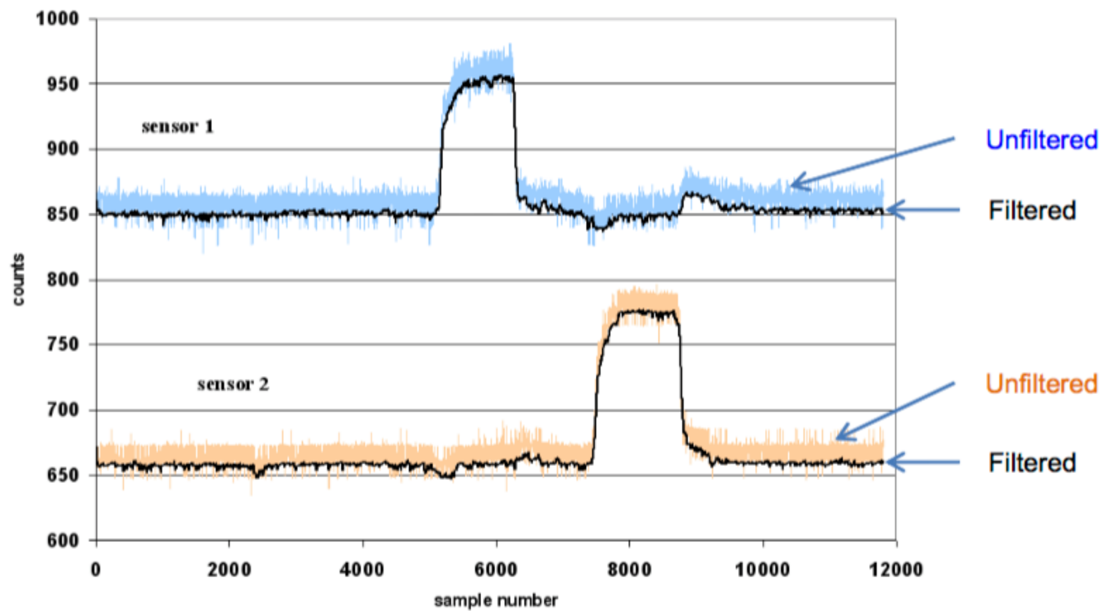
	 Equation 3-3 gives the value of average current supplied to AMUXBUS A. Equation 3-3. Average Current Sourced to AMUXBUS A from GPIO through CapSense Sensor (I_{CS}) $I_{CS} = C_S F_{SW} (V_{DDD} - V_{REF})$ *See* PSoC 4 and PSoC 6 MCU CapSense Design Guide, at p. 19, https://www.cypress.com/file/46081/download
[1c] manipulating the first response and the second response to reject noise at frequencies less than a frequency associated with the pulse.	The CapSense touchcontroller manipulates the first response and the second response to reject noise at frequencies less than a frequency associated with the pulse. For example, CapSense manipulates the row count responses which include the first and second responses (i.e. IDAC sinking and sourcing mode responses) to reject noise having low or at the touch sensor's operational frequency. **Finger Conducted AC Noise** In some cases, low frequency AC noise can affect a circuit when it is operated close to such noise sources. CapSense Express is tested against two such sources. The power system ground of the power supply powering CapSense can inject AC noise of 50 Hz or 60 Hz (depending on the country of use) to CapSense sensor. *See* CAPSENSE(R) EXPRESS(TM) - DESIGN TO PRODUCTION, at p. 10, https://www.cypress.com/file/140271/download

	Figure 3-27. IIR Filter Finger Touch  *See* Getting Started with CapSense, at p. 58, https://www.cypress.com/file/41076/download

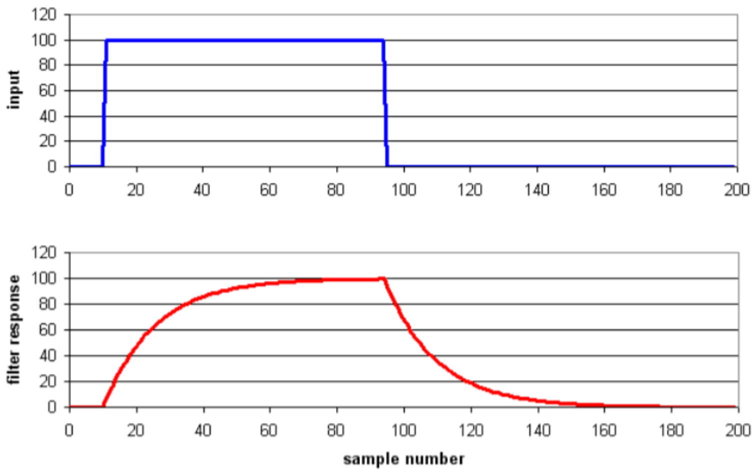
	3.4.2 IIR Filter Infinite impulse response filters (IIR) produce a step response similar to RC filters. IIR filters attenuate high-frequency noise components and pass lower frequency signals, such as finger touch-response waveforms. Figure 3-25. IIR Filter Step Response  The general equation for a first-order IIR filter is: $y[i] = \frac{1}{k}\left(x[i] + ((k-1) \times y[i-1])\right)$ Equation 13 *See* Getting Started with CapSense, at p. 57, https://www.cypress.com/file/41076/download

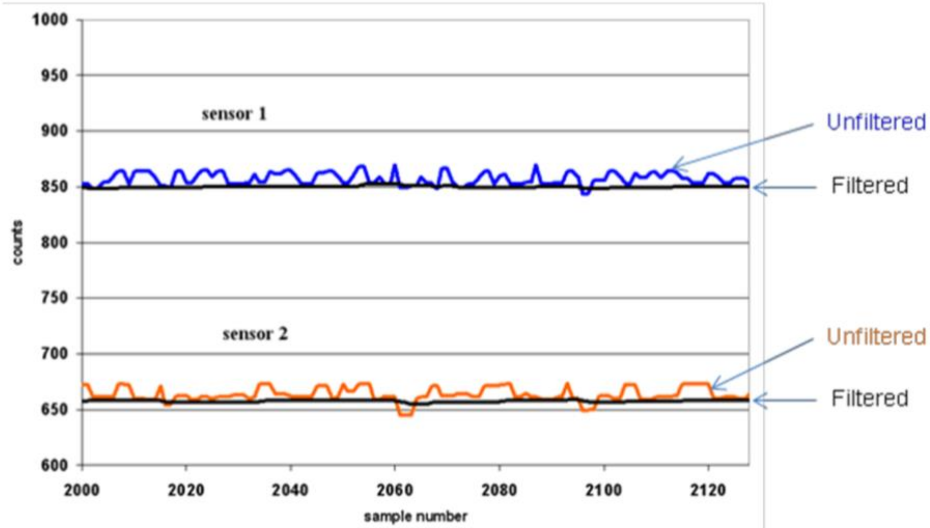
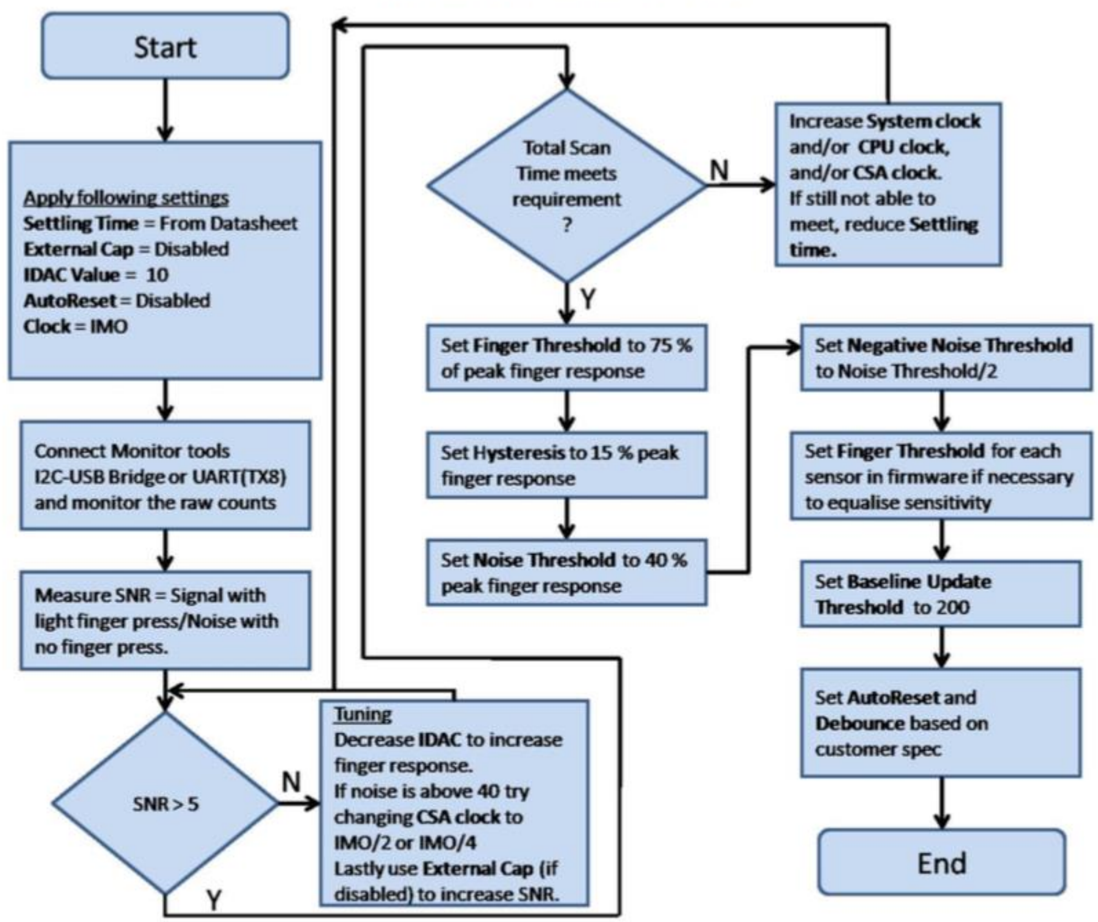
	Figure 3-26 and Figure 3-27 illustrate the results of a first-order IIR filter on real CapSense data using the filter equation with k = 16: $y[i] = \frac{1}{16}\left(x[i] + (15 \times y[i-1])\right)$ Equation 14 Figure 3-26. IIR Filter Noise  *See* Getting Started with CapSense, at p. 57, https://www.cypress.com/file/41076/download

	Figure 8. CSA Calibration Flowchart  *See* CSA Software Filters with Ezl2Cs Slave on CY8C20xx6, at p. 10, https://www.cypress.com/file/137541/download